Efficient Selection of Unique and Popular Oligos for Large EST Databases

Stefano Lonardi
University of California, Riverside

joint work with

Jie Zheng, Timothy Close, Tao Jiang
University of California, Riverside

General problem

- Input: A list of DNA sequences
- Output: A list of short DNA strings of length 20-50 bases (oligos)
  - occur only once in each DNA sequence ("unique" oligos problem)

  or

  - occur in as many DNA sequences as possible ("popular" oligos problem)

Barley genome (*H. vulgare*)

- Size is ~ $5x10^9$ bases
  - 12 times the size of Rice
  - 35 times the size of Arabidopsis
- Too large for whole sequencing
- Strategy
  - Build a BAC library of Barley
  - Identify/sequence only the BACs containing the genes (expected ~ 10%)

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Method

- An EST database for Barley is available
- Use the EST db to identify a set of "popular" oligos that hybridize with as many genes/EST as possible (maximize coverage)
- Use as little oligos/filter/screens as possible (minimize time and money)

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Objectives

- Maximize the *coverage ratio* (number of covered ESTs/number of oligos)
- Minimize the computational resources (memory, time)

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Barley EST db

- Composed by ~ 350K EST sequences
- Cleaned (quality-trimming, cleaned of contaminants, etc.)
- Assembled (pre-clustered, assembled)
- Final dataset (HarvEST v1.07)
  - 46,145 unigenes
  - 28,475,016 bases

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

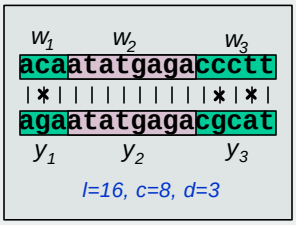
Related work

- Pattern discovery (Meme, Teiresias, Pratt, Gibbs, Projection, Weeder, etc.) cannot be used because of the large input size
- Primer/probe design typically use all-against-all BLAST (eg., [Li&Stormo'01], [Rouillard *et al.*'02]) are extremely slow
- Rahmann [CSB'02] uses suffix arrays (requires ~ 50 hours on Compaq Alpha with 16GB RAM on a dataset of 40Mbases)

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Def: *(c,d)*-match

- Given integers *c* and *d* and strings *w* and *y*, $|w|=|y|$, we say that *w* *(c,d)*-match *y* **iff** *w* and *y* can be partitioned in substrings $w=w_1w_2w_3$ and $y=y_1y_2y_3$ such that
- $|w_1|=|y_1|$ and $|w_3|=|y_3|$
- $w_2=y_2$, $|w_2|=|y_2|=c$ (core)
- $H(w_1w_3,y_1y_3)=d$



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Def: *(c,d)*-coverage

- Given a set $X=\{x_1,\ldots,x_k\}$, a string *y* and integers *c* and *d*, the *(c,d)-coverage* of *y* is the number of sequences of *X* containing each at least one *(c,d)*-match of *y*
- Integer *l* to denote the length of *y (l-mer)*

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

"popular oligos" problem

- Given $X=\{x_1,\ldots,x_k\}$ and integers *l, d, c* and *T*, find all strings of length *l* such that their *(c,d)*-coverage in *X* is *=T*
- We call these strings "popular oligos"
- In our experiments
  l=36, c=20, d=2 or *3, T=2…50*

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Observations

- Note that a popular oligo may never appear exactly in *X*
- Enumerating/counting all $\binom{l-c}{d}(|\Sigma|-1)^d$ possible *(c,d)*-matches of each *l*-mer in *X* is computationally impractical
- For example, if *l=36, c=20*, *d=3,* $|\Sigma|=4$, one should count ~ 15K (20,3)-matches for each 36mer. We have 2*28M 36mers, for a total of 846B elementary operations

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Heuristics: phase one

- Build an hash table for the cores
- For each core w_2 that appears in $=T_c$ (core coverage threshold) sequences
  - Collect all flanking regions w_1w_3, such that $w_1w_2w_3$ is an *l*-mer with popular core w_2
  - Run phase two on set of all extensions w_1w_3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Example: phase one

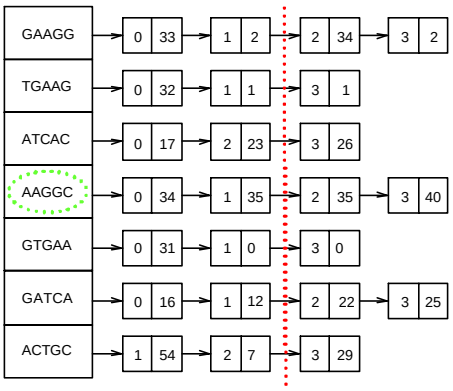
>EST0
TGGAGTCCTCGGACACGATCACATCGACAATGTGAA
GGCGA
>EST1
GTGAAGGAGGTAGATCAAATAGAGCCTGCCCTAAAA
AGGCAGCTTATAATCTCCACTGCT
>EST2
TCCGACTACTGCACCCCGAGCGGATCACACAATGGAA
GGCCCGTGCGC
>EST3
GTGAAGGAGGTAGATACTCGTATACGATCACTGCCTA
AAAAGGCAGCTTATAATCTCCATATCGCTG

l=8, *c*=5, *d*=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Example: phase one

>EST0
TGGAGTCCTCGGACACGATCACATCGACAATGTGAA
GGCGA
>EST1
GTGAAGGAGGTAGATCAAATAGAGCCTGCCCTAAAA
AGGCAGCTTATAATCTCCACTGCT
>EST2
TCCGACTACTGCACCCCGAGCGGATCACACAATGGAA
GGCCCGTGCGC
>EST3
GTGAAGGAGGTAGATACTCGTATACGATCACTGCCTA
AAAAGGCAGCTTATAATCTCCATATCGCTG



l=8, *c*=5, *d*=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Heuristics: phase two (UPGMA)

- Place all $w_1 w_3$ at the leaves of the tree & merge identical leaves
- Build the UPGMA* tree on Hamming distance
- Create a set of *d*-mutants for each string in the leaves of the tree
- Traverse the tree bottom-up performing set intersection
  - as soon as intersection is empty, separate the subtree from the rest of the tree
- The sets at the root of each tree in the forest represent the candidate popular oligo

* Unweighed Pair Group Method with Arithmetic Mean

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Example : phase two (UPGMA)

core **AAGGC**

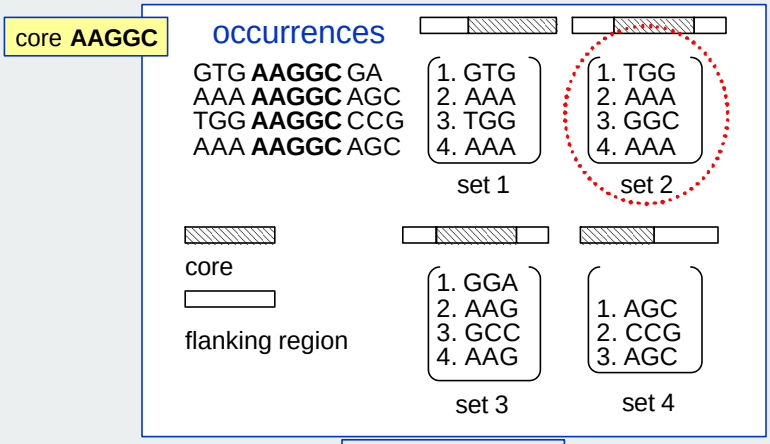
occurrences

GTG **AAGGC** GA
AAA **AAGGC** AGC
TGG **AAGGC** CCG
AAA **AAGGC** AGC

l=8, *c*=5, *d*=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

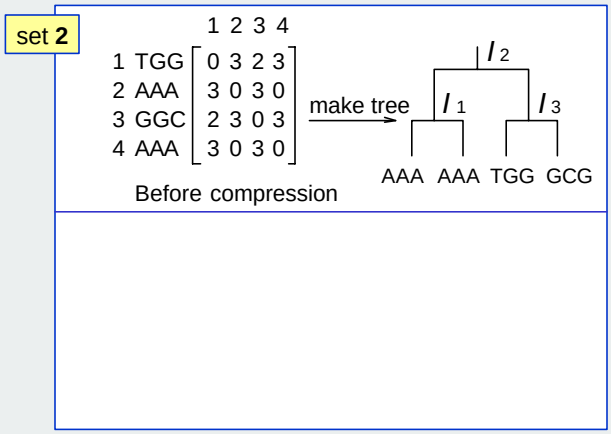
Example : phase two (UPGMA)



l=8, c=5, d=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

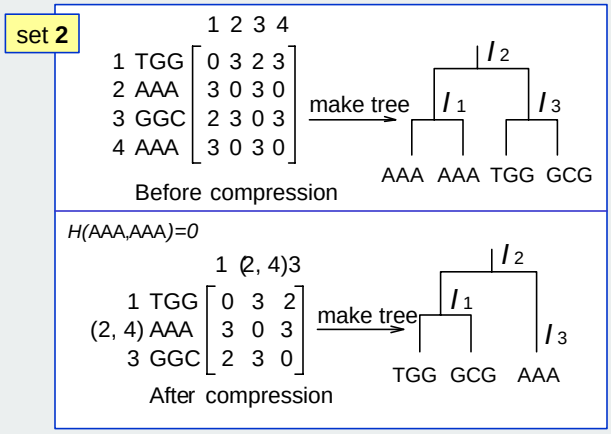
Example : phase two (UPGMA)



l=8, c=5, d=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

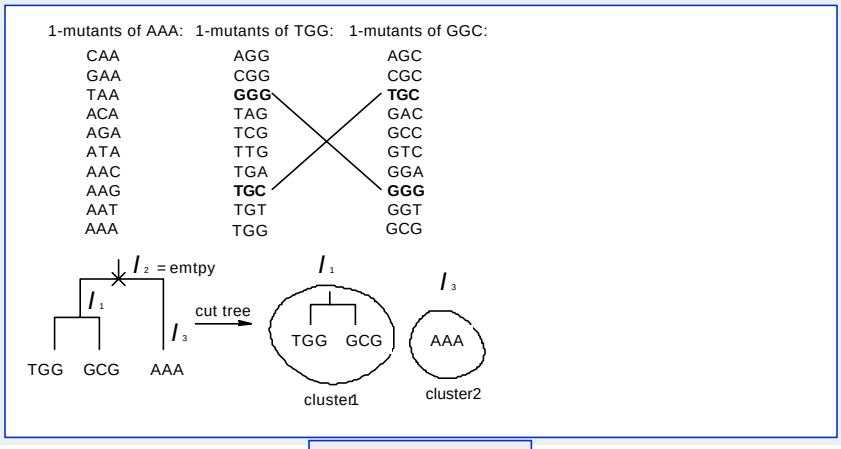
Example : phase two (UPGMA)



l=8, c=5, d=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

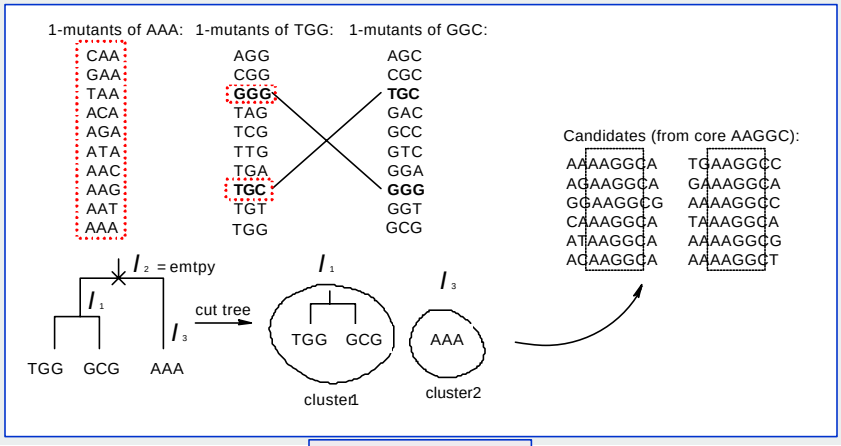
Example : phase two (UPGMA)



l=8, c=5, d=1, T_c=3

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Example : phase two (UPGMA)



l=8, c=5, d=1, T_c=3

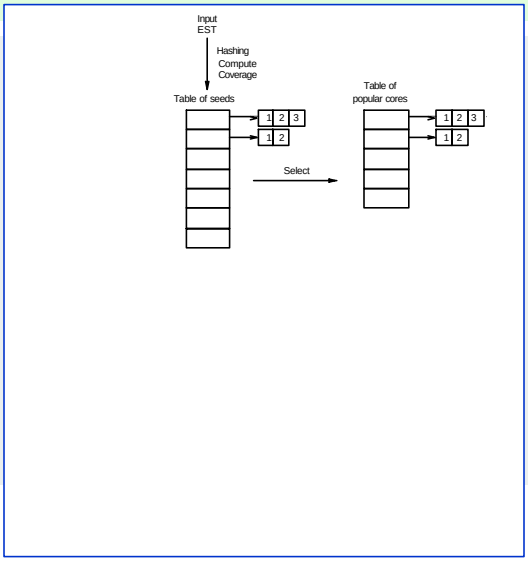
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Heuristics: phase three

- Radix sort the candidate oligos to remove duplicates
- Discard unsuitable oligos
  - low-complexity strings (polyA, polyT, etc.)
  - 44% < GC-content < 56%
- Compute coverage
- Compress/correct oligos

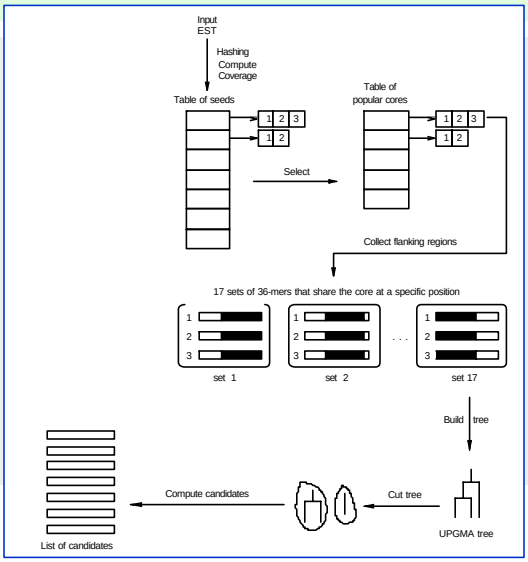
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Overview: phase one



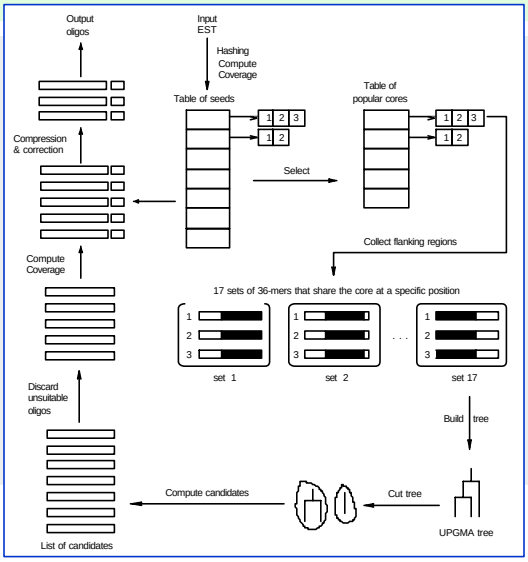
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Overview: phase two (UPGMA)



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Overview: phase three



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Limitation of the heuristics

- Cores which have coverage below T_c are called unpopular
- A *l*-mer can *(c,d)*-match with any of its *l-c+1* cores
- We will miss popular oligos which popularity depend on a combination of several unpopular cores

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Simulations

- Generate $\{x_1,\ldots,x_k\}$ random sequences
- Inject $\{I_1,\ldots,I_s\}$ popular oligos with d errors outside a core of length c, with coverage $\{C_1,\ldots,C_s\}$ (Gaussian distribution, max coverage R)
- Run the popular oligo algorithm on $\{x_1,\ldots,x_k\}$

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Simulation

- Obtain $\{O_1,\ldots,O_t\}$ with coverage $\{C'_1,\ldots,C'_s\}$ (sorted)
- $\{O_1,\ldots,O_t\}$ is compressed
- Compare (I,C) with (O,C')
- For each $1=i=u$ for $u=min(s,t)$ we compute

$$E(C,C') = \frac{1}{u}\sum_{i=1}^{u}\frac{|C_i - C_i'|}{C_i'}$$

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Simulation results

- *k=2000, $|x_i|$=720, c=20, s=100, R=100*

*E*100*	*d=2*	*d=3*
T_c*=10*	1.89	5.30
T_c*=15*	0	0.07
T_c*=20*	0.6	0.06
T_c*=25*	0	0.31
T_c*=30*	0.07	0.25

- We never miss any oligo whose coverage is above T_c*+10*

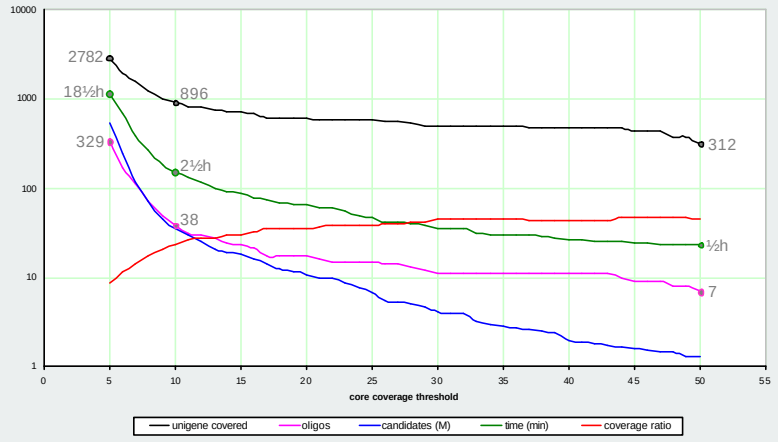
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Experimental results

- *l=36* (oligo length)
- *c=20* (core length)
- *d=2,3* (max mismatches outside core)
- T_c*=varies* (core coverage threshold)
- *k=46,145* unigenes
- *n=28* million bases
- PC with 1.2 GHz CPU and 1GB memory

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Coverage graph



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Current & Future work

- Progressive processing to reduce memory requirements
- Fine tuning & optimization of the code
- New strategies to improve coverage ratio
- New definition for popular/unique oligos
- Parallel implementation

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

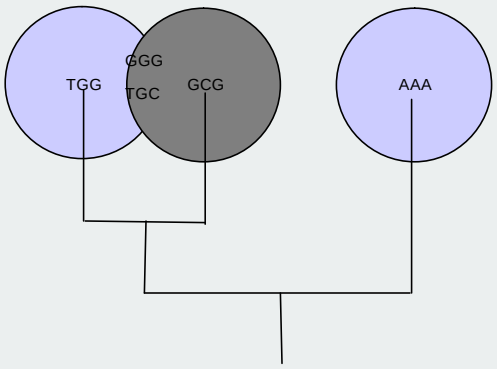
Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

Complexity

- Build a seed table *O(cn)*
- Collect flanking substrings *O(nr(l-c))* where *r* is # occurrences of cores
- Building UPGMA $O\left(r\binom{l-c}{d}3^d\right)$
- Counting colors for *m* candidate *O(rm(l-c))*

Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside

UPGMA + intersection



Stefano Lonardi
Department of CS and E
Bourns College of Engineering
University of California, Riverside